**LAB CYCLE 1**

1. Program to Print all non-Prime Numbers in an Interval.

Code:

def is\_prime(num):

if num <= 1:

return False

for i in range(2, num):

if num % i == 0:

return False

return True

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("-------------------------------------------------------------------------------")

N = int(input("Enter the number :"))

current\_num = 2

print( end=" ")

while N > 0:

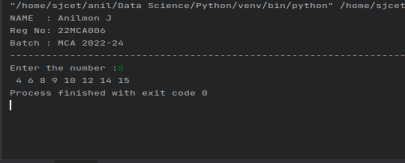
if not is\_prime(current\_num):

print(current\_num, end=" ")

N -= 1

current\_num += 1

Output:



2. Program to print the first N Fibonacci numbers.

Code:

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("--------------------------------------------------------------------------------")

num = int(input("Enter the value of n: "))

a= 0

b = 1

n = 0

count = 1

while(count <= num):

print(n,end=" ")

count += 1

a = b

b = n

n = a + b

t\_number = a + b

Output:



3. Given sides of a triangle, write a program to check whether given triangle is an isosceles, equilateral or scalene.

Code:

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("--------------------------------------------------------------------------------")

print("Input length of Triangle Sides:")

x=int(input("x:"))

y=int(input("y:"))

z=int(input("z:"))

if( x == y == z):

print("Equilateral triangle")

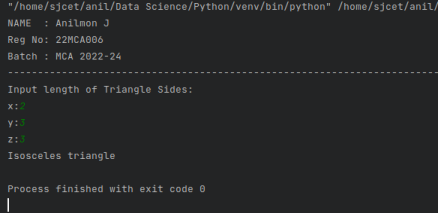
elif(x == y or y==z or z==x):

print("Isosceles triangle")

else:

print("Triangle is Scalene")

Output:



4. Program to check whether given pair of number is coprime.

Code:

import math

def are\_coprime(a, b):

gcd = math.gcd(a, b)

return gcd == 1

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("--------------------------------------------------------------------------------") a = int(input("Enter the first number: "))

b= int(input("Enter the second number: "))

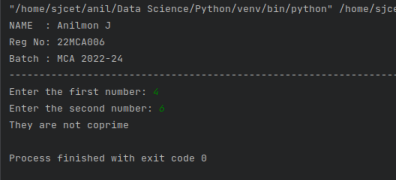
if are\_coprime(a,b):

print(f"{a} and {b} are coprime")

else:

print("They are not coprime")

Output:



5. Program to find the roots of a quadratic equation(rounded to 2 decimal places).

Code:

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("------------------------------------------------------------------------------") import math

a = float(input("Enter value of a: "))

b = float(input("Enter value of b: "))

c = float(input("Enter value of c: "))

discri = b\*\*2 - 4\*a\*c

if discri > 0:

root1 = (-b + math.sqrt(discri)) / (2\*a)

root2 = (-b - math.sqrt(discri)) / (2\*a)

print(f"Root 1: {round(root1, 2)}")

print(f"Root 2: {round(root2, 2)}")

elif discri == 0:

root = -b / (2\*a)

print(f"Root: {round(root, 2)}")

else:

real\_part = -b / (2\*a)

img\_part = math.sqrt(-discri) / (2\*a)

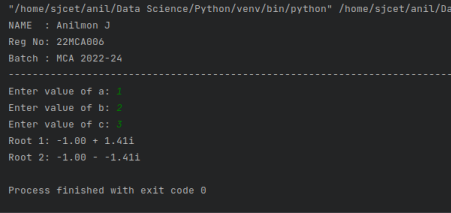
root1 = complex(real\_part, img\_part)

root2 = complex(real\_part, -img\_part)

print(f"Root 1: {root1.real:.2f} + {root1.imag:.2f}i")

print(f"Root 2: {root2.real:.2f} – {root2.imag:.2f}i")

Output:



6. Program to check whether a given number is perfect number or not(sum of factors=number).

Code:

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("--------------------------------------------------------------------------------")

n=int(input("Enter a number:\n"))

sum=0

for i in range(1,n):

if(n % i == 0):

sum=sum+i

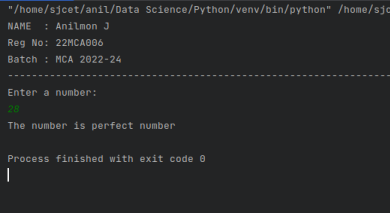
if( sum == n):

print("The number is perfect number")

else:

print("The entered number is not perfect")

Output:



7. Program to display amstrong numbers upto 1000.

Code:

def is\_armstrong\_number(num):

num\_str = str(num)

num\_digits = len(num\_str)

armstrong\_sum = sum(int(digit) \*\* num\_digits for digit in num\_str) return armstrong\_sum == num

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

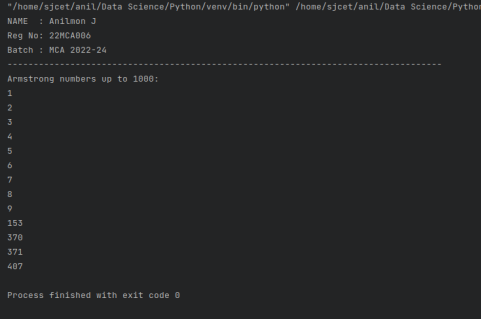
print("--------------------------------------------------------------------------------") print("Armstrong numbers up to 1000:")

for num in range(1, 1001):

if is\_armstrong\_number(num):

print(num)

Output:



8. Store and display the days of a week as a List, Tuple, Dictionary, Set. Also demonstrate different ways to store values in each of them. Display its type also.

**Code:**

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("--------------------------------------------------------------------------------") days\_list = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"]

print("List:", days\_list)

print("Type:", type(days\_list))

days\_tuple = ("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday")

print("Tuple:", days\_tuple)

print("Type:", type(days\_tuple))

days\_dict = {0: "Monday", 1: "Tuesday", 2: "Wednesday", 3: "Thursday", 4: "Friday", 5: "Saturday", 6: "Sunday"}

print("Dictionary:", days\_dict)

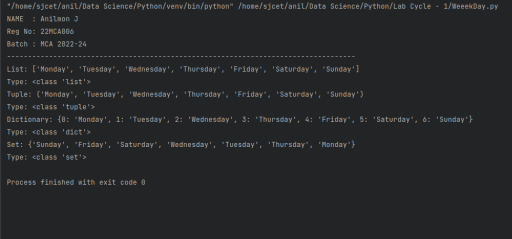
print("Type:", type(days\_dict))

days\_set = {"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"}

print("Set:", days\_set)

print("Type:", type(days\_set))

Output:



9. Write a program to add elements of given 2 lists.

Code:

def add\_lists(a,b):

if len(a) != len(b):

return None

result = []

for i in range(len(a)):

result.append(a[i] + b[i])

return result

try:

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("--------------------------------------------------------------------------------") a = input("Enter the first list of numbers : ").split()

a = [int(x) for x in a]

b = input("Enter the second list of numbers : ").split()

b = [int(x) for x in b]

result = add\_lists(a, b)

if result is None:

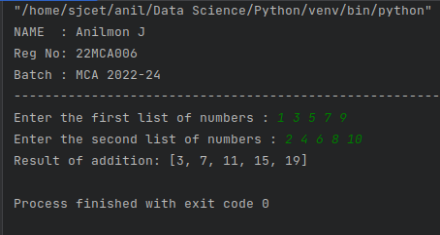
print("The lists have different lengths and cannot be added.") else:

print("Result of addition:", result)

except ValueError:

print("Invalid input. Please enter valid numbers separated by spaces.")

Output:



10. Write a program to find the sum of 2 matrices using nested List.

**Code:**

def add\_matrices(matrix1, matrix2):

if len(matrix1) != len(matrix2) or len(matrix1[0]) != len(matrix2[0]): return None

result = [[0 for \_ in range(len(matrix1[0]))] for \_ in range(len(matrix1))]

for i in range(len(matrix1)):

for j in range(len(matrix1[0])):

result[i][j] = matrix1[i][j] + matrix2[i][j]

return result

try:

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("--------------------------------------------------------------------------------") rows = int(input("Enter the number of rows: "))

cols = int(input("Enter the number of columns: "))

print("Enter elements of the first matrix:")

matrix1 = []

for i in range(rows):

row = input(f"Enter elements of row {i + 1} separated by spaces: ").split()

matrix1.append([int(x) for x in row])

print("Enter elements of the second matrix:")

matrix2 = []

for i in range(rows):

row = input(f"Enter elements of row {i + 1} separated by spaces: ").split()

matrix2.append([int(x) for x in row])

result = add\_matrices(matrix1, matrix2)

if result is None:

print("Matrix dimensions are not compatible for addition.") else:

print("Sum of matrices:")

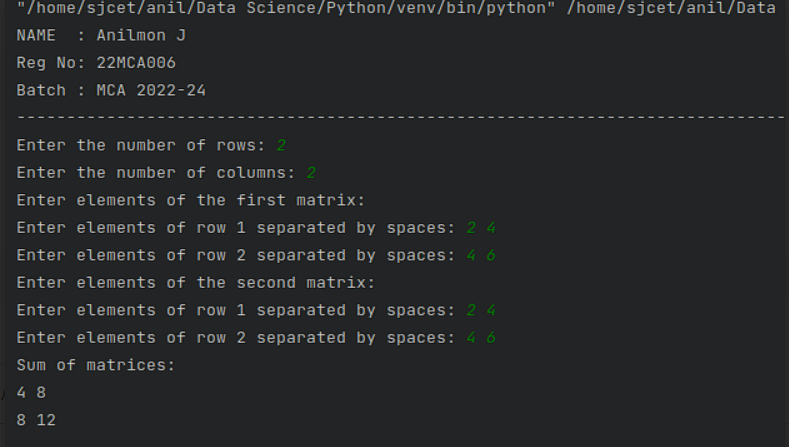
for row in result:

print(" ".join(map(str, row)))

except ValueError:

print("Invalid input. Please enter valid numbers.")

Output:



11. Write a program to perform bubble sort on a given set of elements.

Code:

def bubble\_sort(arr):

n = len(arr)

for i in range(n):

swapped = False

for j in range(0, n - i - 1):

if arr[j] > arr[j + 1]:

arr[j], arr[j + 1] = arr[j + 1], arr[j]

swapped = True

if not swapped:

break

try:

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("--------------------------------------------------------------------------------") elements = input("Enter elements separated by spaces: ").split() elements = [int(x) for x in elements]

bubble\_sort(elements)

print("Sorted elements:")

print(elements)

except ValueError:

print("Invalid input. Please enter valid numbers separated by spaces.")

Output:



12. Program to find the count of each vowel in a string(use dictionary)

Code:

def count\_vowels(string):

vowel\_counts = {'A': 0, 'E': 0, 'I': 0, 'O': 0, 'U': 0}

string = string.upper()

for char in string:

if char in vowel\_counts:

vowel\_counts[char] += 1

return vowel\_counts

try:

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("--------------------------------------------------------------------------------") input\_string = input("Enter a string: ")

vowel\_counts = count\_vowels(input\_string)

print("Vowel counts:")

for vowel, count in vowel\_counts.items():

print(f"{vowel}: {count}")

except ValueError:

print("Invalid input. Please enter a valid string.")

Output:



13. Write a Python program that accept a positive number and subtract from this number the sum of its digits and so on. Continues this operation until the number is positive(eg: 256->2+5+6=13

256-13=243

243-9=232……..

Code:

def sum\_of\_digits(n):

digit\_sum = 0

while n > 0:

digit\_sum += n % 10

n //= 10

return digit\_sum

try:

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("--------------------------------------------------------------------------------") num = int(input("Enter a positive number: "))

if num <= 0:

print("Please enter a positive number.")

else:

while num > 0:

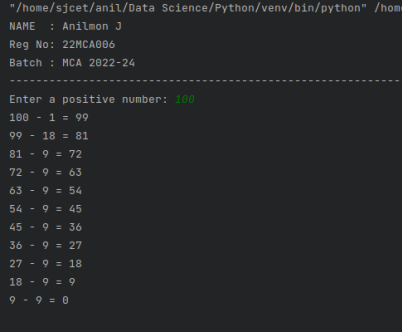
digit\_sum = sum\_of\_digits(num)

print(f"{num} - {digit\_sum} = {num - digit\_sum}") num -= digit\_sum

except ValueError:

print("Invalid input. Please enter a valid positive number.")

Output:



14. Write a Python program that accepts a 10 digit mobile number, and find the digits which are absent in a given mobile number.

Code:

def find\_absent\_digits(mobile\_number):

all\_digits = set("0123456789")

mobile\_digits = set(mobile\_number)

absent\_digits = all\_digits - mobile\_digits

return sorted(list(absent\_digits))

try:

print("NAME : Anilmon J")

print("Reg No: 22MCA006")

print("Batch : MCA 2022-24 ")

print("--------------------------------------------------------------------------------") mobile\_number = input("Enter a 10-digit mobile number: ")

if len(mobile\_number) == 10 and mobile\_number.isdigit(): absent\_digits = find\_absent\_digits(mobile\_number)

if absent\_digits:

print("Absent digits in the mobile number:", ', '.join(absent\_digits)) else:

print("The mobile number contains all digits from 0 to 9.") else:

print("Invalid input. Please enter a valid 10-digit mobile number.") except ValueError:

print("Invalid input. Please enter a valid 10-digit mobile number.")

Output:

